

WHAT IS CLAIMED IS:

1. A method for dynamically modifying a stored program, the method comprising the steps of:

5 storing correction code in at least one of a plurality of correction blocks included in an electrically erasable programmable memory;

executing a program having instructions stored in the memory;

invoking an address match routine to execute at least a portion of the correction code in place of at least one of the instructions during the executing of the program;

10 and

continuing executing the program after the at least a portion of the correction code executes.

2. The method of claim 1, wherein the step of invoking an address match routine occurs when a program counter associated with the executing of the program matches at least one of a plurality of address match registers.

3. The method of claim 2, further comprising the steps of:

20 determining for each of the correction blocks whether correction code stored in the blocks is to be executed during the executing of the program;

retrieving a first address from the correction code stored in a respective correction block when it is determined that correction code stored in the respective correction block is to be executed during program execution; and

25 storing the retrieved first address in one of the plurality of address match registers.

4. The method of claim 3, wherein the step of determining for each of the correction blocks whether correction code stored in a respective correction block is to be executed during program execution comprises the steps of:

30 retrieving a first data value from each of the correction blocks having stored correction code;

comparing the retrieved first data value from each of the correction blocks having stored correction code to a predetermined value; and

determining that the correction code stored in each of the correction blocks is to be executed during program execution when the corresponding first data value equals the predetermined value, otherwise determining that the correction code stored in each of the correction blocks is not to be executed during program execution.

5. The method of claim 2, wherein the step of determining for each of the correction blocks whether correction code stored in the blocks is to be executed during the executing of the program occurs in response to a completion of the storing of correction code in at least one of a plurality of correction blocks.

6. The method of claim 2, further comprising the steps of:
saving a plurality of registers and the program counter upon the invoking of the address match routine to preserve a post address match program status;
retrieving a second data value from each of the correction blocks having stored correction code;
comparing the retrieved second data value from each of the correction blocks having stored correction code to a post address match value of the program counter;
and

identifying the correction block corresponding to the invoking of the address match routine as the correction block having the second data value equal to the post address match value of the program counter.

7. The method of claim 6, further comprising the step of:
branching to an error processing routine when no retrieved second data value is equal to the post address match value of the program counter.

8. The method of claim 6, wherein the second data value is equal to the value stored in the address match register corresponding to the invoking of the address match routine plus an offset value.

9. The method of claim 8, wherein the offset value is dependent upon a next program instruction to be executed at the time of the invoking of the address match routine.

5 10. The method of claim 6, wherein the step of continuing executing the program after the at least a portion of the correction code executes comprises the steps of:

retrieving a third address from the correction code identifying a return address within the program;

10 restoring the plurality of registers saved upon the invoking of the address match routine to reinstate a post address match program status;

branching the executing of the program to the third address; and

executing the program having instructions stored in the memory beginning at the third address.

11. The method of claim 1, wherein the invoking of the address match routine. is carried out by an address match interrupt service routine having a corresponding address match interrupt entry in a vector table stored in the memory.

120 12. The method of claim 1, wherein to execute at least a portion of the correction code, the address match routine comprises the steps of:

retrieving a second address from the correction code identifying a starting address within the at least a portion of correction code;

branching the executing of the program to the second address; and

25 executing the at least a portion of the correction code beginning at the second address.

30 13. The method of claim 1, wherein instructions for continuing executing the program after the at least a portion of the correction code executes are included in the correction code.

106290-08796860

14. The method of claim 1, wherein the step of storing correction code in at least one of a plurality of correction blocks comprises the steps of:

selecting at least one of the plurality of correction blocks for storing the correction code;

5 erasing the selected at least one memory correction block; and

transferring the correction code from an external source to the selected at least one of the plurality of correction blocks.

10 15. The method of claim 14, wherein the correction code is transferred from the external source by at least one of a wired and a wireless connection.

16. The method of claim 1, wherein the step of storing correction code in at least one of a plurality of correction blocks occurs in response to at least one of a detection that an external source is coupled to the memory and an invocation of a periodically scheduled maintenance routine.

17. A micro-controller capable of dynamically modifying a stored program, the micro-controller comprising:

electrically erasable programmable memory;

20 logic that stores correction code in at least one of a plurality of correction blocks included in the memory;

logic that executes a program having instructions stored in the memory;

25 logic that invokes an address match routine to execute at least a portion of the correction code in place of at least one of the instructions during the executing of the program; and

logic that continues executing the program after the at least a portion of the correction code executes.

30 18. The micro-controller of claim 17, wherein the logic that invokes an address match routine is responsive to a matching of a program counter associated with the executing of the program and at least one of a plurality of address match registers.

logic that determines for each of the correction blocks whether correction code stored in the blocks is to be executed during the executing of the program;

logic that retrieves a first address from the correction code stored in a respective correction block when it is determined that correction code stored in the respective correction block is to be executed during program execution; and

logic that stores the retrieved first address in one of the plurality of address match registers.

20. The micro-controller of claim 19, wherein the logic that determines for each of the correction blocks whether correction code stored in a respective correction block is to be executed during program execution comprises:

logic that retrieves a first data value from each of the correction blocks having stored correction code:

logic that compares the retrieved first data value from each of the correction blocks having stored correction code to a predetermined value; and

logic that determines that the correction code stored in each of the correction blocks is to be executed during program execution when the corresponding first data value equals the predetermined value, otherwise determining that the correction code stored in each of the correction blocks is not to be executed during program execution.

21. The micro-controller of claim 18, wherein the logic that determines for each of the correction blocks whether correction code stored in the blocks is to be executed during the executing of the program is responsive to a completion of the storing of correction code in at least one of a plurality of correction blocks.

22. The micro-controller of claim 18, further comprising:

logic that saves a plurality of registers and the program counter upon the invoking of the address match routine to preserve a post address match program status;

logic that retrieves a second data value from each of the correction blocks having stored correction code;

1
 2
 3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525

logic that compares the retrieved second data value from each of the correction blocks having stored correction code to a post address match value of the program counter; and

logic that identifies the correction block corresponding to the invoking of the address match routine as the correction block having the second data value equal to the post address match value of the program counter.

23. The micro-controller of claim 22, further comprising:

logic that branches to an error processing routine when no retrieved second data value is equal to the post address match value of the program counter.

24. The micro-controller of claim 22, wherein the second data value is equal to the value stored in the address match register corresponding to the invoking of the address match routine plus an offset value.

25. The micro-controller of claim 24, wherein the offset value is dependent upon a next program instruction to be executed at the time of the invoking of the address match routine.

26. The micro-controller of claim 22, wherein the logic that continues executing the program after the at least a portion of the correction code executes comprises:

logic that retrieves a third address from the correction code identifying a return address within the program;

logic that restores the plurality of registers saved upon the invoking of the address match routine to reinstate a post address match program status;

logic that branches the executing of the program to the third address; and

logic that executes the program having instructions stored in the memory beginning at the third address.

27. The micro-controller of claim 17, wherein the logic that invokes the address match routine is responsive to an address match interrupt service routine

032674-140-062901

having a corresponding address match interrupt entry in a vector table stored in the memory.

28. The micro-controller of claim 17, further comprising:

5 logic that retrieves a second address from the correction code identifying a starting address within the at least a portion of correction code;
logic that branches the executing of the program to the second address; and
logic that executes the at least a portion of the correction code beginning at the second address.

10 29. The micro-controller of claim 17, wherein instructions for continuing executing the program after the at least a portion of the correction code executes are included in the correction code.

15 30. The micro-controller of claim 17, wherein the logic that stores correction code in at least one of a plurality of correction blocks comprises:
logic that selects at least one of the plurality of correction blocks for storing the correction code;
logic that erases the selected at least one memory correction block; and
20 logic that transfers the correction code from an external source to the selected at least one of the plurality of correction blocks.

25 31. The micro-controller of claim 30, wherein the correction code is transferred from the external source by at least one of a wired and a wireless connection.

30 32. The micro-controller of claim 17, wherein the logic that stores correction code in at least one of a plurality of correction blocks is responsive to at least one of a detection that an external source is coupled to the memory and an invocation of a periodically scheduled maintenance routine.

33. The micro-controller of claim 17, wherein the electrically erasable programmable memory is included on at least one of a same electronic chip as the remaining logic comprising the micro-controller and a different electronic chip as the remaining logic comprising the micro-controller.

5

34. An electrically erasable programmable memory based memory map structure supporting an address match interrupt scheme, the memory comprising:

- a plurality of correction blocks for storing correction code;
- a random access memory (RAM) area including a program stack for temporarily

10 storing program information;

- a main program area for storing at least one executable program;
- an initialization area for storing code to enable an address match interrupt for at least one of the correction blocks;
- a special function register (SFR) area including a plurality of address interrupt

5 registers;

- a vector table for triggering the address match interrupt when a program counter associated with the at least one executable program matches a register value stored in one of the plurality of address interrupt registers; and
- an interrupt service routine (ISR) area including an address match ISR;

20 wherein the address match ISR identifies which one of the plurality of correction blocks corresponds to the triggering of the address match interrupt to execute at least a portion of the correction code in place of at least one instruction of the at least one executable program during program execution.

25 35. The structure of claim 34, wherein each of the correction blocks comprise:

- a first data value for determining whether correction code stored in the correction block is to be executed during program execution;
- a second data value for identifying which one of the plurality of correction blocks corresponds to the triggering of the address match interrupt;

30 a first address identifying a location of the portion of the executable program replaced by the at least a portion of the correction code;

a second address identifying a starting address of the at least a portion of the correction code to be executed in place of the least a portion of the executable program; and

- 5 a third address identifying a return address within the executable program where program execution is continued after execution of the at a portion of the correction is completed.

T06290-08/96860